Listing of Claims:

1. (Cancelled)

2. (Currently Amended) The method of claim 1~~3~~ wherein the first set of operation tasks includes navigating existing data structure links.

3. (Currently Amended) The method of claim 1~~3~~ wherein the step of developing a second set of operation tasks further comprises developing ~~the~~ a set of pointers to the data structure.

4. (Currently Amended) The method of claim 1~~3~~ wherein the first phase comprises performing parallel operations on the linked data structure and the second phase comprises performing serial operations on the linked data structure, each of the serial operations being developed during one of the parallel operations.

5. (Currently Amended) The method of claim ~~1~~ 3 wherein the set of pointers is stored in a list.

6. (Previously Presented) The method of claim 5 wherein the set of pointers is stored in a first in last out list.

7. (Currently Amended) The method of claim 1~~3~~ wherein the step of developing a second set of operation tasks further comprises a step of performing a conflicts check ~~for the operation~~.

8. (Currently Amended) ~~The method of claim 1 wherein the first set of element state transitions further comprises:~~A method for executing an operation upon a linked data structure having at least one element, the method comprising the steps of:

performing a first set of operation tasks in a first phase, the first set of operation tasks operable to effect a first set of element state transitions comprising

(a) a valid state to a pending delete state transition~~;~~,

(b) a pre-associated state to a pending insert state transition~~;~~, and

(c) a pending insert state to a hidden state transition;

developing a second set of operation tasks, the second set of operation tasks operable to effect a second set of element state transitions and being associated with a set of pointers to the linked data structure, the set of pointers being stored external to the linked data structure, the second set of element state transitions being distinct from the first set of element state transitions; and

performing the second set of operation tasks in a second phase using the set of pointers.

9. (Currently Amended) A method for executing an operation upon a linked data structure having at least one element, the method comprising the steps of:

performing a first set of operation tasks in a first phase, the first set of operation tasks operable to effect a first set of element state transitions;

developing a second set of operation tasks, the second set of operation tasks operable to effect a second set of element state transitions and being associated with a set of pointers to the linked data structure, the set of pointers being stored external to the linked data structure, the second set of element state transitions being distinct

from the first set of element state transitions~~The method of claim 1, wherein the second set of element state transitions further~~ and ~~comprises~~comprising~~:~~

a pending insert state to a valid state transition~~;~~,

(a) a pending delete state to an invalid state transition~~;~~,

(b) a hidden state to an invalid state transition~~;~~,

(c) a pending delete state to a valid state transition,~~;~~

(d) a hidden state to a pending insert state transition~~;~~, and

(e) a pending insert state to an invalid state transition; and

performing the second set of operation tasks in a second phase using the set of pointers.

10. – 12. (Cancelled)

13. (Previously Presented) A method of inserting a plurality of elements into a linked data structure comprising the steps of:

(a) performing a first set of operation tasks in a first phase, the first set of operation tasks operable to effect a first set of element state transitions including a pre-associated state to a pending insert state transition for each of the plurality of elements;

(b) developing a second set of operation tasks, the second set of operation tasks operable to effect a second set of element state transitions including a pending insert state to a valid state transition for each of the plurality of elements; and

(c) performing the second set of operation tasks in a second phase.

14. (Previously Presented) The method of claim 13 wherein the pre-associated state to a pending insert state transition is accomplished by:

(a) marking an element to be inserted as being pre-associated to the data structure;

(b) navigating the data structure to an insertion point;

(c) creating links between the element to be inserted and the data structure at the insertion point, the links created being visible only to the insertion operation; and

(d) marking the element as being pending insert.

15. (Original) The method of claim 14 wherein the pending insert state to a valid state transition is accomplished by:

(a) creating instructions for making the created links visible to all operations; and

(b) creating instructions for making existing links at the insertion point invisible to all operations.

16. (Original) The method of claim 15 wherein the step of performing the second set of operation tasks further comprises executing the created instructions including marking the element as valid.

17. (Previously Presented) A method of deleting a plurality of elements from a linked data structure comprising the steps of:

(a) performing a first set of operation tasks in a first phase, the first phase including a valid state to a pending delete state transition for each of the plurality of elements;

(b) developing a second set of operation tasks, for execution in a second phase including a pending delete state to an invalid state transition for each of the plurality of elements; and

(c) performing the second set of operation tasks in the second phase.

18. (Original) The method of claim 17 wherein the valid state to a pending delete state transition is accomplished by:

(a) navigating the data structure to a deletion point;

(b) creating links at the deletion point visible only to the deletion operation; and

(c) marking the element to be deleted as pending delete.

19. (Original) The method of claim 18 wherein the pending delete state to an invalid state transition is accomplished by:

(a) creating instructions for making the created links visible to all operations; and

(b) making existing links at the deletion point invisible to all operations.

20. (Previously Presented) The method of claim 19 wherein the step of performing the second set of operation tasks further comprises executing the instructions including marking members of the plurality of elements to be deleted as invalid.

21. – 23. (Cancelled)

24. (Currently Amended) The method of claim ~~23~~ 13 wherein the first set of operation tasks is operable in parallel to maintain the linked data structure in an existing linked state.

25. (Currently Amended) The method of claim ~~24~~ 13 wherein the second set of operation tasks are operable in series to modify ~~the~~ an existing linked state.

26. (Currently Amended) The method of claim ~~23~~ 17 wherein each of the first set of operation tasks is visible only to one of ~~the~~ a plurality of operations.

27. (Previously Presented) The method of claim 26 wherein the second set of operation tasks is visible to each of the plurality of operations.

28. – 36. (Cancelled)

37. (Currently Amended) The method of claim ~~36~~ 13 wherein ~~at least one of~~ the first set of~~simultaneous~~ operation tasks ~~includes an element insertion operation, the first phase operation task of the element insertion operation being~~are performed on an unlocked portion of the linked data structure.

38. (Currently Amended) The method of claim ~~36~~ 13 wherein ~~at least one of~~ the ~~simultaneous operations includes an element deletion operation, the second phase operation task of the element deletion operation being~~second set of operation tasks are performed independently of navigation of the linked data structure.

39. (Currently Amended) The method of claim ~~36~~13 wherein the first ~~phase~~ set of operation tasks are asynchronous and use existing links to navigate the linked data structure.

40. – 42. (Cancelled)

43. (Currently Amended) The method of claim ~~41~~17, wherein developing the ~~plurality of~~ second set of ~~phase~~ operation tasks includes developing a series of pointers to elements of the linked data structure and storing the series of pointers external to the linked data structure.

44. (Currently Amended) The method of claim ~~41~~17, wherein the ~~plurality of~~ first ~~phase~~ set of operation tasks are performed in parallel on the linked data structure in an unlocked state and the ~~plurality of~~ second ~~phase~~ set of operation tasks are performed in series.

45. (Currently Amended) The method of claim ~~41~~17, wherein the ~~plurality of~~ second ~~phase~~ set of operation tasks are performed by navigating to a plurality of elements within the linked data structure using a plurality of pointers external to the linked data structure.

46. – 47. (Cancelled)

48. (Previously Presented) A method of inserting a plurality of elements into a linked data structure, the method comprising:

in a first phase

navigating the linked data structure to a first insertion point for insertion of a first member of the plurality of elements,

associating the first member of the plurality of elements with the linked data structure, the first member of the plurality of elements being in a pending insert state,

storing a first pointer to the first insertion point in a list external to the linked data structure,

navigating the linked data structure to a second insertion point for insertion of a second member of the plurality of elements,

associating the second member of the plurality of elements with the linked data structure, the second member of the plurality of elements being in a pending insert state,

storing a second pointer to the second insertion point in the list external to the linked data structure, storing the first pointer and storing the second pointer; and

in a second phase separate from the first phase

reading the first pointer from the list,

using the read first pointer to navigate to the first insertion point,

completing the insertion of the first member of the plurality of elements in the linked data structure,

reading the second pointer from the lines,

using the read second pointer to navigate to the second insertion point, and

completing the insertion of the second member of the plurality of elements in the linked data structure.

49. (New) A computer readable medium having stored therein:

a code segment configured for performing a first set of operation tasks in a first phase, the first set of operation tasks operable to effect a first set of element state transitions including a pre-associated state to a pending insert state transition for each of the plurality of elements;

a code segment configured for developing a second set of operation tasks, the second set of operation tasks operable to effect a second set of element state

transitions including a pending insert state to a valid state transition for each of the plurality of elements; and

a code segment configured for performing the second set of operation tasks in a second phase, to insert a plurality of element into the linked data structure.

50. (New) A computer readable medium having stored therein:

a code segment configured for performing a first set of operation tasks in a first phase, the first phase including a valid state to a pending delete state transition for each of the plurality of elements;

a code segment configured for developing a second set of operation tasks, for execution in a second phase including a pending delete state to an invalid state transition for each of the plurality of elements; and

a code segment configured for performing the second set of operation tasks in the second phase, to delete a plurality of elements from a linked data structure.

51. (New) A computer readable medium having stored therein:

a code segment configured for performing a first set of operation tasks in a first phase, the first set of operation tasks operable to effect a first set of element state transitions comprising

(a) a valid state to a pending delete state transition,

(b) a pre-associated state to a pending insert state transition, and

(c) a pending insert state to a hidden state transition;

a code segment configured for developing a second set of operation tasks, the second set of operation tasks operable to effect a second set of element state transitions and

being associated with a set of pointers to a linked data structure, the set of pointers being stored external to the linked data structure, the second set of element state transitions being distinct from the first set of element state transitions; and

a code segment configured for performing the second set of operation tasks in a second phase using the set of pointers.

52. (New) A computer readable medium having stored therein:

a code segment configured for performing a first set of operation tasks in a first phase, the first set of operation tasks operable to effect a first set of element state transitions;

a code segment configured for developing a second set of operation tasks, the second set of operation tasks operable to effect a second set of element state transitions and being associated with a set of pointers to a linked data structure, the set of pointers being stored external to the linked data structure, the second set of element state transitions being distinct from the first set of element state transitions, and comprising

(e) a pending insert state to a valid state transition,

(f) a pending delete state to an invalid state transition,

(g) a hidden state to an invalid state transition,

(h) a pending delete state to a valid state transition,

(i) a hidden state to a pending insert state transition, and

(f) a pending insert state to an invalid state transition; and

a code segment configured for performing the second set of operation tasks in a second phase using the set of pointers.